ОБ ОДНОМ ПОДХОДЕ К ПРЕДСТАВЛЕНИЮ ДАННЫХ В СИСТЕМЕ «ВИЗОГРАФ»

Е.А. Козинов

Нижегородский госуниверситет им. Н.И. Лобачевского

Разработка параллельных программ является сложной задачей. Одно из активно развиваемых направлений, уменьшающих сложность, – создание сред визуального проектирования и разработки параллельных программ. В рамках статьи рассматривается одна из таких систем – «Визограф». Для системы рассматривается один из подходов к описанию типов данных, позволяющий контролировать как синтаксическую, так и логическую правильность получаемых визуальных схем.

Введение

На протяжении многих лет одной из наиболее острых проблем эффективного использования потенциала компьютерных технологий является трудоемкость разработки программного обеспечения (ПО). Широкий спектр практических приложений, для решения которых разрабатываются программы, разнообразие компьютерных систем и их непрерывное совершенствование, существенное повышение сложности применяемых алгоритмов обработки – все это вместе взятое чрезвычайно затрудняет разработку ПО. В отличие от компьютерного оборудования, производительность которого практически удваивается каждые 18 месяцев (широко известное утверждение Мура), технологии разработки программ совершенствуются крайне медленно – к сожалению, закон Мура для ПО не действует.

Для параллельных вычислительных систем проблема трудоемкости разработки становится критической. Дополнительные сложности вызывает относительная новизна области параллельного программирования, наличие качественно новых эффектов при выполнении вычислительных процессов (тупики, гонки потоков, избыточная синхронизация и т.п.). Особая острота проблемы связана с необходимостью оперативной массовой разработки параллельных программ вследствие коренного перехода компьютерной техники на использование многоядерных процессоров.

Одно из активно развиваемых направлений, уменьшающих сложность, – создание сред визуального проектирования и разработки параллельных программ [1]. Смысл подхода заключается в том, что разрабатываемый алгоритм разделяется на независимые части. Каждая часть представляется в виде компонента или модуля. Для получения параллельного алгоритма модули компонуются по средствам визуальных схем.

Визуальная система «Визограф»

В Нижегородском государственном университете им. Н.И. Лобачевского разрабатывается новая среда визуального проектирования и разработки параллельных программ «Визограф» [2]. Основные усилия направлены на решение следующих основных задач:

1. Поиск и разработка методов улучшения визуального (наглядного) представления разрабатываемых параллельных алгоритмов и программ.
2. Расширение визуального подхода средствами автоматической верификации и контроля правильности построенных вычислительных схем.
3. Разработка – наряду с возможностями «ручного» конструирования визуальных схем – автоматических или полуавтоматических средств поддержки пользователя в разработке алгоритмов для решения поставленной задачи.

В разрабатываемой системе для представления алгоритмов используются ряд визуальных элементов. Основными элементами являются модули, параметры и линкеры. Каждый модуль соответствует некоторой реализации определенного алгоритма или его части и может быть представлен, например, в виде программного модуля dll-библиотеки. Параметры используются для визуального представления входных и выходных данных модулей. Линкеры применяются для установления соответствия (связывания) между параметрами модулей. Подобные элементы встречаются в большинстве визуальных систем разработки параллельных программ, использующих визуальные элементы для отображения потока данных [1]. Примеры модулей можно увидеть на рис. 1, блоки 2 и 6.

Одно из основных назначений визуальных схем – упрощение процесса разработки алгоритма. Если модулей, выделенных в алгоритме, много, то за счет большого количества связей визуальная схема становится сложной для восприятия. Для упрощения представления алгоритма, как правило, применяется иерархическое представление. В системе «Визограф» иерархическое представление обеспечивается путем введения макромодулей (см. рис. 1, блок 1).

Для повышения компактности изображения алгоритмов в системе «Визограф» дополнительно введены новые элементы «сплиттер» и «агрегатор». Благодаря этим элементам, данные можно разделить на части для последующей параллельной обработки и объединить результаты вычислений. Пример сплиттера и агрегатора можно увидеть на рис. 1, блоки 3 и 5.

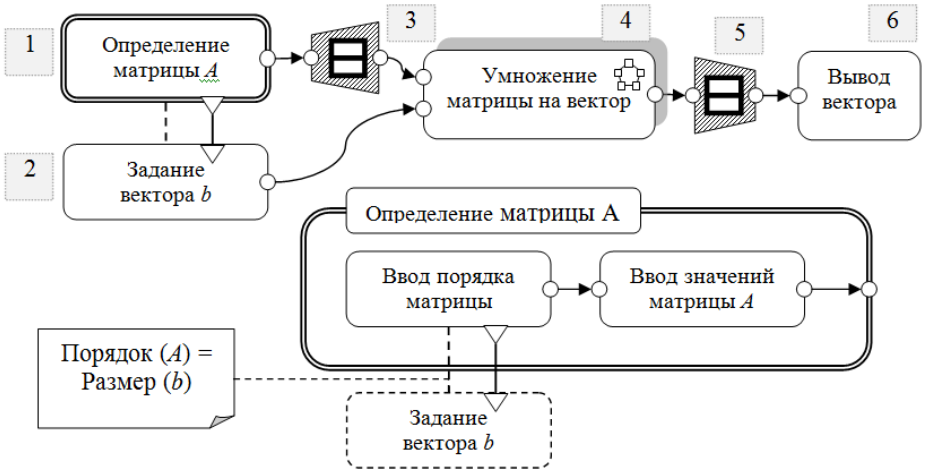

Рис. 1. Основные элементы системы «Визограф» на примере алгоритма умножения матрицы на вектор

Для повышения контроля над данными в процессе исполнения в визуальной системе «Визограф» введены визуальные элементы обозначающие параметры связности. На рис. 1. благодаря введенным элементам, отображено и контролируется одно из условий – для перемножения матрицы на вектор порядок матрицы A должен совпадать с размером вектора b.

Более подробную информацию о визуальных элементах, используемых во время разработки алгоритмов, и методах их применения можно получить в [2]. Далее рассматривается один из возможных методов представления данных в системе «Визограф».

Представление типов данных в системе «Визограф»

В разрабатываемой системе «Визограф» предполагается, что схема корректна, если выполнен ряд условий. Одно из таких условий: все параметры модулей должны быть соединены с помощью линкеров (циклические связи при этом не допускаются).

Для обеспечения синтаксической корректности построенных схем недостаточно того, чтобы все параметры в построенной схеме были соединены посредством линкеров. Введенные параметры связности позволяют лишь частично проверить логическую правильность построенных схем. Основное же их назначение – помочь в отладке получаемых параллельных программ в процессе исполнения. Для обеспечения синтаксической корректности построенной схемы необходимо, чтобы параметры, соединенные линкерами, содержали данные одного типа. Для представления типов данных в системе «Визограф» введен ряд дополнительных визуальных элементов.

На начальном этапе использования системы «Визограф» пользователю доступны только *базовые типы данных*. Предполагается, что параметры модулей могут быть сопоставлены только с данными типа матриц, векторов или скаляров. Сами базовые типы данных могут содержать элементы разных типов данных, таких как целые (integer), вещественные (double, float) и др. Для обозначения типов данных используются прямоугольники с пиктограммами в правом верхнем углу. Пиктограмма визуально обозначает используемый базовый тип данных. Внутри прямоугольников явно указывается тип элементов. Пример базового типа данных – матрица вещественного типа – приведен на рис. 2, блок 1.

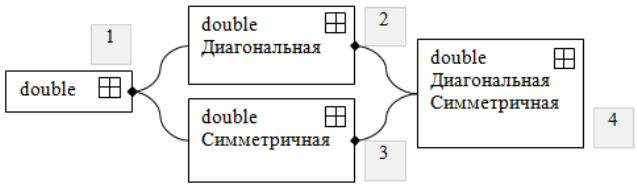

Рис. 2. Представление типов данных в системе «Визограф»

При разработке алгоритмов часто не хватает информации о том, что на вход модуля поступили данные того или иного типа. Для выполнения многих алгоритмов необходимо, что бы данные обладали определенным набором свойств. Например, разрабатываемый алгоритм может быть рассчитан на то, что используется матрица диагональная или симметричная. Пример таких задач легко найти, например, среди методов решения систем линейных уравнений.

Для разработки сложных алгоритмов необходимо иметь возможность включения автоматической проверки корректности принимаемых и получаемых данных. Подобная проверка позволяет сконцентрировать усилия на разработке самого алгоритма, а не проверке входных данных.

Для проверки корректности данных в системе «Визограф» введены *производные типы данных*. Производный тип данных может быть получен путем добавления свойств из системы «Визограф» к существующему типу данных.

Необходимые свойства могут быть добавлены в систему «Визограф» благодаря разработке макромодуля, удовлетворяющего ряду требований. Данный макромодуль через входящий параметр должен принимать базовый или уже существующий тип данных. Через исходящий параметр макромодуль в виде скаляра, содержащего логическое значение, должен возвращать результат проверки свойства. Любые свойства, реализуемые пользователем, сохраняются в системе и могут быть повторно использованы.

Во время исполнения алгоритма можно включить проверку свойств. В этом случае при получении и передаче данных для каждого параметра выполняется запуск алгоритмов проверки свойств. Алгоритмы проверки свойств вызываются из макромодулей и могут исполняться параллельно.

Изображение производных типов похоже на изображение базовых типов. В изображение, кроме типа элементов, добавляется список свойств. Пример определения двух производных типов от матрицы изображен на рис. 2, блоки 2 и 3.

В некоторых случаях необходимо получить тип данных путем агрегации свойств уже существующих типов данных. Для предоставления такой возможности в системе «Визограф» поддерживается агрегация типов данных. Свойства агрегируемых типов данных объединяются. Пример агрегированного типа данных изображен на рис. 2, блок 4.

Логическое описание типов данных в системе «Визограф»

При разработке алгоритмов совпадение типов данных параметров, соединенных посредством линкеров, не гарантирует правильность работы всего алгоритма в целом. Корректная визуальная схема с точки зрения синтаксиса не гарантирует корректности работы алгоритма с логической точки зрения. Логику алгоритма нельзя описать только физическими свойствами параметров. Для описания логики нужны дополнительные метасвойства, поясняющие назначение параметров. В качестве таких метасвойств в системе «Визограф» используются *дескрипторы*.

Дескрипторы добавляются в типы данных так же, как и свойства. Все дескрипторы отображаются в списке свойств. Для отделения физических свойств от дескрипторов типов данных дескрипторы отображаются после вертикальной черты. При агрегации типов данных множество дескрипторов, заключенных в типах данных, также объединяется. Дескрипторы являются глобальными для системы «Визограф» и могут быть повторно использованы.

На рис. 3, блок 1, представлен пример определенного типа данных – матрица вещественного типа, используемая в операциях умножения матрицы. Дополнительные примеры использования дескрипторов для описания логического назначения определяемых типов данных представлены на рис. 3, блоки 2 и 3.

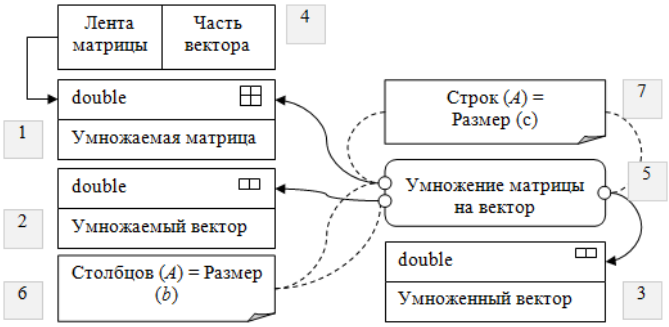

Рис. 3. Пример регистрации модуля в системе «Визограф»

Не редка ситуация, когда однажды разработанный модуль может быть применен для разработки частей алгоритмов, использующих разные с логической точки зрения наборы параметров, но физически реализующий один и тот же алгоритм. Пример такого модуля – умножение матрицы на вектор. Алгоритм умножения матрицы на вектор не изменяется ни в случае, когда умножается матрица на вектор, ни в случае, когда умножается лента матрицы на вектор. Отличие состоит в том, что при умножении ленты матрицы на вектор в результате получается только часть результирующего вектора.

Для описания логических альтернатив применения типов данных в системе «Визограф» вводится понятие *дерево альтернативных дескрипторов*. Если для типа данных есть возможность добавить дополнительные логические описания, которые расширяют возможность применения типа данных, то это изображается в виде прямоугольника, разделенного на две части вертикальной линией. В левой части прямоугольника перечисляются дескрипторы, добавляемые к типу данных. В правой части перечисляются дескрипторы, которые автоматически добавятся к каждому типу данных, соответствующему исходящему параметру модуля, использующему тип данных с учетом расширения. Для связывания подобных структур используются линкеры. Иерархия альтерна-

тивных дескрипторов в системе «Визограф» не ограничена. Каждый новый уровень добавляет логические свойства к получаемому типу данных. Пример альтернативного дескриптора изображен на рис. 3, блок 4.

В системе «Визограф» присутствуют особые элементы – сплиттеры и агрегаторы. Для поддержания логической связности схем слиттеры и агрегаторы автоматически добавляют и удаляют дескрипторы, связанные с разделением или объединением данных.

Введенное логическое описание, добавляемое к типам данных, закладывает основы для автоматического построения схем. Благодаря дескрипторам, пользователь системы может сформулировать решаемую задачу и предоставить системе возможность помочь найти решение в виде схемы алгоритма.

Регистрация модулей в системе «Визограф»

Базовые типы данных, свойства и дескрипторы позволяют сконструировать необходимые типы данных. Далее построенные типы данных используются для регистрации модулей в системе.

Общий процесс регистрации модуля в системе проходит в несколько этапов. Вначале пользователь системы конструирует необходимые или выбирает наиболее подходящие среди существующих типы данных. Затем инструментальными средствами создается пустой модуль. При создании модуля ему задается имя. Далее к модулю добавляется необходимое количество входящих и исходящих параметров. Для каждого параметра задается имя. Затем параметры модуля связываются с типами данных, содержащими логическое и синтаксическое описания. Выполнив данную последовательность действий, модуль можно зарегистрировать в системе. На рис. 3, блок 5, представлен зарегистрированный модуль умножения матрицы на вектор.

Важно заметить, что при регистрации модулей для параметров можно указать параметры связности. Указание параметров связности позволяет снизить вероятность появления ошибок на ранних этапах разработки. На рис. 3, блоки 6 и 7, показан пример добавления параметров связности.

Зарегистрированные модули могут быть использованы для разработки алгоритмов. При связывании модулей контролируется логическая и синтаксическая правильность построенных схем.

Заключение

В Нижегородском государственном университете им. Н.И. Лобачевского разрабатывается новая среда визуального проектирования и разработки параллельных программ «Визограф». В рамках статьи рассматривается один из подходов к описанию типов данных, используемых в системе. Представленное описание позволяет осуществлять автоматическую верификацию и контроль правильности построенных вычислительных схем. Также представленное описание потенциально может быть применено для автоматической или полуавтоматической поддержки пользователя в разработке алгоритмов для решения поставленных задач.

Литература

1. Гергель В.П., Козинов Е.А. Обзор визуальных систем разработки параллельных программ // Вестник ННГУ им. Н.И. Лобачевского. [принята к печати]
2. Гергель В.П., Козинов Е.А. Об одном подходе к визуальной разработке программ // Вестник ННГУ им. Н.И. Лобачевского. 2012. № 4. С. 247-252.